Chapter 1: Introduction

Abstract

TART95 is a coupled neutron-photon Monte Carlo transport code. This computer code is designed to use three dimensional (3-D) combinatorial geometry. Neutron and/or photon sources as well as neutron induced photon production can be tracked by this code. The code can calculate: 1) static reactivity problems, 2) dynamic reactivity problems, 3) source problems involving any combination of neutron and/or photon sources. The code allows for a very wide variety of source descriptions, as well as scoring and output options that meet the needs of most applications; this is one of the most powerful features of TART95 that make it so versatile for use in such a wide variety of applications.

TART95 is now available for use on virtually any computer. It has already been implemented and used on CRAY, HP, SUN, SGI, Meiko, DEC-Alpha, IBM-RSIC, and IBM-PC. It is written in such standard FORTRAN that it should be relatively easy to implemented and use on any computer.

Dedication

The report is dedicated to the memory of Sterrett T. Perkins who passed away in March, 1995. For the last quarter century Ted was a major contributor to all aspects of our efforts in particle transport, from evaluating nuclear and atom data, all the way through to the models used in all of our transport method, including Monte Carlo, S_n and diffusion. He was both a great theorist and yet at the same time a very practical person; an attribute that is very rare to find in a person. He was also a good friend and he will be sorely missed by all of us.

Acknowledgments

Over the years many people contributed to the development of the TART family of codes. There are many people who should be acknowledged and thanked for their efforts. The following is a partial list in which we hope we do not overlook anyone. We acknowledge and thank, Richard Bentley, Donald Davis, William Grayson, Robert Howerton, John Terrall, Robert Wright, and most of all John Kimlinger who spent more than sixteen years working on the TART family of codes. On a more recent note, we acknowledge and thank Jeff Latkowski who significantly contributed in testing the implementation of TART95 on workstations; many thanks Jeff, we couldn't have done it without you. Last, but certainly not least, we thank Susan Mangels for helping to format and produce the final version of this report.

Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract number W-7405-ENG-48.

Overview of this Report

The report is not designed as a general text on Monte Carlo; there are more than enough publications on this subject. It is only intended as a user manual for the TART95 code and as such only discusses those methods used by TART95, in sufficient detail to allow users to understand what the code is doing and how to use it.

An important consideration in designing this report was to insure that it would be available to users on-line on computers. Due to the current incompatibility between various word processors, particularly with respect to complicated equations and figures, it was decided to constrain this report to contain only text, using Microsoft Word. Hopefully the result will be a report that can be translated and used by users who have one of any number of other word processors that can read Microsoft Word text files.

For on-line use this report has been divided into a number of self contained chapters, each containing its own table of contents and subject index. The pages have been numbered independently within each chapter to allow each chapter to be maintained, updated, and distributed independently of the other chapters.

The only portion of this report that is not available on-line is the figures. However, the distributed code, its utilities, and example problems, will allow the interested user to re-produce these figures on their own computers, particularly all those figures produced by the interactive graphics utility code TARTCHEK.

We sincerely hope that this approach meets your needs, and we would appreciate any comments (pro or con) concerning this report and its usefulness in your applications.

Introduction

TART95 is a coupled neutron-photon Monte Carlo transport code. This computer code is designed to use three dimensional (3-D) combinatorial geometry. Neutron and/or photon sources as well as neutron induced photon production can be tracked by this code. The code can calculate: 1) static reactivity problems, 2) dynamic reactivity problems, 3) source problems involving any combination of neutron and/or photon sources. The code allows for a very wide variety of scoring and output options that meet the needs of most applications; this is one of the most powerful features of TART95 that make it so versatile for use in such a wide variety of applications.

One of the biggest advantages of TART95 compared to other neutron-photon Monte Carlo codes is speed. If you compare the running time of a variety of Monte Carlo codes running the same calculations you may be surprised to find problems that take others codes hours or many minutes to complete can be run in a few minutes or even seconds using TART95.

TART95 achieves this speed by building on the almost 40 years of experience using the TART family of codes [1]; hundreds of man-years of user experience using methods that experience has shown to be both fast and accurate. TART95's treatment of geometry and nuclear and atom data, to only include what's important (only to within the accuracy that it is known) and necessary in calculations in order to quickly obtain accurate answers, particularly contribute to the speed of the code.

Another advantage of TART95 is that it is simple to use; you do not have to be an expert in particle transport and Monte Carlo variance reduction to use TART95. You do have to learn to prepare input parameters defining your problem's geometry, sources, and what you would like to be output, etc. But beyond this point you will be able to successfully use TART95 and it will use its built-in expertise to apply variance reduction and other techniques to get you an accurate answer as fast as it can.

Regardless of how good, fast and accurate a code is, if it cannot produce the types of results that you are interested in it isn't much use to you. As far as the advantages of TART95, we should again mention the wide variety of scoring and output options that meet the needs of most applications. The most basic output is energy dependent results for each zone, e.g., current across zone boundaries, fluence and energy deposition within zones. However, TART95 will allow you to tally and output results as a function of space (by zone), energy, direction, and even time. For example, if you would like to know the leakage from a system as a function of energy, direction (the angular distribution), and time, just ask for it. We encourage users to spend the time to become familiar with the variety of scoring and output options available with TART95.

TART95 is designed to be as similar as possible to the TARTNP code [1], as far as the user interface, e.g., it uses the same input parameters and produces similar output results. This program differs from TARTNP in that TARTNP was originally written in LRLTRAN, a dialect of FORTRAN developed at Livermore and generally incompatible with the FORTRAN used elsewhere. TARTNP was designed to be used on the large central computers available at Livermore; currently CRAY computers, and is still heavily used at Livermore. In contrast TART95 is written in modern FORTRAN and is designed to be used on large central computers as well as a wide variety of currently available workstations and even small personal computers, e.g., IBM-PCs.

The TARTNP code is quite an old code, which give it both disadvantages and advantages. Its disadvantages include that it was originally designed when the available computers were relatively small, forcing design compromises in order to implement the code. It was also originally written in the now outdated LRLTRAN language and is very poorly documented, both in terms of documentation to allow users to use the code and in terms of documentation to allow programmers to maintain and update the code. Its advantages are that for a wide variety of applications it is still the fastest neutron-photon Monte Carlo transport code available, and what should not be overlooked is that it now has hundreds of man-years of user and programmer experience incorporated into it.

In designing TART95 we have attempted to keep the advantages of TARTNP and eliminate its disadvantages. In particular TART95 is now written in modern FORTRAN that has already allowed it to be easily implemented on a wide variety of computers. In addition the size of the code has been expanded, in line with the large size of currently available workstations, to allow it to handle larger, more complicated problems. Most important, by using much of the original logic from TARTNP, the speed of the code has been not only maintained, but improved. Carrying over much of the logic of TARTNP has also allowed much of the experience of TARTNP to also be incorporated in the new TART95 code. However, the user must appreciate that TART95 is a new code and as such it will take some time for it to establish a track record similar to TARTNP's, as far as reliability and user acceptance. As such users should not consider TART95 to be an immediate replacement for the production version of TARTNP, as currently used on the Livermore CRAYs, but rather as a step toward providing a reliable replacement for TARTNP before the Livermore CRAYs are replaced and the current production version of TARTNP is no longer available. Therefore we encourage TARTNP users to try TART95 as soon as possible, so that we may establish a track record using your applications. If you follow this advice there should be a smooth transition of your work from CRAY to workstations, without encountering any inconveniences to you or delay in accomplishing your work.

Constraints on the Design of TART95

TART95 is designed to eventually replace the CRAY production version of TARTNP. TART95 is designed to be used on as many different types of computers as possible; it has presently been implemented and tested on CRAY, SUN, SGI, HP, Meiko, DEC-Alpha, IBM-RSIC, and IBM-PC. At the same time the initial version of TART95 is designed to produce results that are as close as possible to the results obtained using the CRAY production version of TARTNP.

In line with this last constraint, TART95 uses the same nuclear and atomic data files used by TARTNP, as well as the methods used to interpret the data in these files; no attempt has been made to improve the neutron and photon data used by both TARTNP and TART95. TART95 also uses the same user interface as TARTNP; both input parameters and output results are in the same form for both codes. This approach has simplified testing and verification of the new TART95 code by allowing it to be directly compared to TARTNP using a wide variety of test problems. It has also made it much easier for users who are familiar with TARTNP to shift over to using TART95, e.g., if you are used to using TARTNP and have any decks of input parameters you can run exactly the same decks using TART95 and the results that you obtain will be in the same format.

Emphasis in designing this first version of TART95 has been to develop a code that does the same things as the existing CRAY production version of TARTNP, but can be used on virtually any computer. Extending the initial version of TART95 to do other types of calculations, or to use different nuclear or atomic data, or to use different methods was not a priority. The top priority was to convert TARTNP from the outdated LRLTRAN language to modern FORTRAN, eliminating as much computer dependence as possible. Even this modest objective required that most of the original TARTNP code be replaced, so that although the user interface looks the same to users, the code that is actually being run is new and completely different from TARTNP.

Only after TART95 has been more widely used and accepted will the code be extended. Some areas that could be extended include: 1) a more general geometry treatment to include cubic and quartic surfaces, e.g., a torus, 2) a more modern treatment of nuclear and atomic data, e.g., the treatment of photon kinematics by TARTNP is from the 1960's and while adequate could be easily updated and improved, similarly many of the neutron evaluations are now quite old and could easily be replaced by currently available modern evaluations, 3) the TARTNP 175 neutron groups, that has now been used for 20 years, could be extended to include more groups to take advantage of the current much larger computers, 4) extension to transport other particles, such as electrons, positrons and charged particles; this extension would be in line with development of the All Particle Transport Method, 5) an improved user interface both in terms of assisting in preparation of input parameters and in terms of assisting in interpreting results, e.g., use codes to go directly from blueprints to TART95 input parameters.

Benchmark Results

If you search the literature you will find literally hundreds of reports documenting the ability of TARTNP to accurately reproduce benchmark results for a wide variety of applications. We have no intention of duplicating the contents of all of these reports here. Any benchmark results presented in this report have a different aim than to prove the accuracy and wide applicability of TARTNP. Based on published results we here assume that TARTNP is accurate. The aim of benchmarking in this report is much simpler and in line with the design criteria and constraints described above: since we assume TARTNP is accurate, in this report all we have to do is demonstrate that TART95 results agree with TARTNP results. If we can do this much simpler task we will have met our goal of producing a workstation code that agree with TARTNP results.

Are Workstations Practical?

Traditionally using large 3-D Monte Carlo codes have required large central computers. Is it really practical to use a workstation or even an IBM-PC to perform these types of calculations? The answer is yes. Today workstations and even IBM-PCs have grown in memory size and speed to the point where they are very competitive, and in some cases faster, than large central computers.

Later in this report we present example results running 68 fast critical assemblies using both the old TARTNP code, that can only be used on CRAY computers, and TART95 using a variety of computers. Here we present the time required to run these calculations. Running time is still not a standard FORTRAN feature on all computers. In order to implement TART95 on many different computers we were forced to use whatever running time information we could obtain on each computer. Therefore the times presented here are in some cases central processor times (CPU) and in other cases wall clock time; CPU time will always be equal to or less than wall clock time. However we should stress that the times for both CRAY results are CPU times, so that the below timing results are the best running times that a CRAY-YMP can produce for this example problem.

Code	Computer	Running Time (Seconds)	Ratio to TARTNP CRAY-YMP
TARTNP	CRAY-YMP	5396	1.0
TART95	CRAY-YMP	4912	0.91
TART95	HP-350	4322	0.80
TART95	DEC-Alpha	6130	1.14
TART95	SUN	9673	1.79
TART95	Meiko	9993	1.85
TART95	SGI	10157	1.88
TART95	IBM-RSIC	14838	2.75
TART95	IBM-PC	18437	3.41

These results illustrate several important points,

1) Workstations and even an IBM-PC are now competitive with large central computers. Using exactly the same TART95 code on a CRAY-YMP and HP-350 shows that the HP-350 is about 12 % faster than the CRAY-YMP. Even an IBM-PC is competitive in the sense that the CRAY-YMP times are CPU times while the IBM-PC is wall clock time. Due to time sharing on the CRAY-YMP, the IBM-PC actually completed the calculations in less wall clock time and we had the results on our desk faster; when we consider people's salaries getting results quickly is a major concern.

2) TARTNP has always been known as a very fast code. Comparison of the TARTNP and TART95 results on a CRAY-YMP illustrates that in converting TART95 for use on workstations, not only have we maintained this speed advantage, we have improved it. By comparing the CRAY-YMP results we can see that the newer TART95 code runs about 10 % faster than the older TARTNP code.

Bottom line on are workstations practical: you bet they are!!!!

The History of the TART Family of Codes

The present TARTNP and TART95 codes are built upon almost 40 years of experience in Monte Carlo transport at Lawrence Livermore National Laboratory. During this time we estimate that 250 to 300 man-years of programmer and user experience have been accumulated and incorporated into the present codes.

Shortly after Lawrence Livermore National Laboratory opened in 1953 work began on the first generation of Livermore Monte Carlo codes. Between 1953 and 1962 much of the methodology, techniques and conversion of software from machine language to the earliest versions of FORTRAN was accomplished. During these years a number of specialized Monte Carlo codes were developed, that contributed to the generation of more generalized codes that were to follow. By 1960 work had begun on the SORS code [2-4].

The first production version of the SORS code in 1962, the direct predecessor of TART, was a Monte Carlo neutron transport code; it did not transport photons. It used the Livermore Evaluated Nuclear Library (ENDL) cross sections, but not the kinematics from the ENDL library; SORS used hardwired models for secondary energy and angular distributions. By today's standards the kinematics were crude, but adequate for their day. Simple hardwired temperature models were used, only one isotropically scattering inelastic level was allowed, elastic angular distributions used simple models to predict their energy dependence. This meant that the results were sensitive to changes in the basic ENDL cross sections, but users could not fully take advantage of the rapid improvements in evaluated nuclear data during the early 1960's and subsequent years.

Preceding the development of TART, the TORTE code was developed in the later 1960's to transport photons. TORTE provided a wide variety of user requested scoring options which made the code extremely versatile. The basic transport and scoring (tally) structure of TORTE was later used in TART. The extensive experience gained in using TORTE significantly contributed to reducing the time and energy required to develop TART and later TARTNP.

Based on the experience gained with SORS and TORTE, the original TART code in 1972 [2-4] was designed to be totally data driven. There were no longer any hardwired data or models in the code. With this approach any improvements in the evaluated nuclear data in ENDL could be immediately seen in the results of calculations. In addition TART could now be used to improve the basic evaluated data by allowing direct comparisons between the most recent experimental measurements and the results of TART Monte Carlo calculations. This step of "closing the loop" between experiments, evaluations and calculations led to rapid improvements in both the basic evaluated data in ENDL and in the agreement between a wide variety of experimental and calculational results. Of particular note is the improvements based on the Livermore pulsed sphere measurements [16-21]. Starting from 14 MeV neutrons at the origin of each sphere these measurements examined the time dependent leakage of neutrons from a wide variety of materials across the periodic table; roughly 40 combinations of material and sphere thickness were used in these measurements. The results of these measurements were instrumental in improving the understanding of basic nuclear processes, as well as allowing a systematic evaluation of what processes were important to model in order to obtain acceptable agreement between experimental measurements and Monte Carlo calculations. One result of these pulsed sphere measurement was the understanding of how important it is to include correlated energy-angular distributions, which resulted in the inclusion of this data in ENDL evaluated neutron data and simultaneous developments of methods for inclusion in TART to use this data in Monte Carlo calculations. It's fair to say that TART was well ahead of all the other Monte Carlo transport codes that are only now facing the problem of considering the new correlated energy-angular distributions included in the most recent ENDF/B-VI library.

During 1973-74 TART was extended to include both neutron induced photon production and photon transport cross sections and kinematics. Once again the TORTE transport and scoring modules were used and TARTNP (TART Neutron-Photon) was born. As in the case of the development of TART, the development of TARTNP led to rapid improvements in the Livermore Evaluated Data, since coupled neutron-photon transport calculations required improved representations of the details of the physical processes involved; this was particularly true with respect to the division between local energy deposition and photon production due to neutron interactions to obtain realistic comparisons to experiments while maintaining strict energy conservation.

Until about 1975 TARTNP was used mostly for higher energy applications; this was mostly due to its use of 175 multi-group neutron cross sections. The multi-group treatment meant that TARTNP could not properly handle resonance self-shielding, nor did it have any treatment of thermal scattering in terms of either Doppler broadening of the cross sections or the actual kinematics of elastic thermal scattering. In 1975 TARTNP was upgraded to include the multi-band method [26-31] to handle resonance self-shielding and a free atom scattering model to handle thermal scattering both in terms of Doppler broadening cross sections using the SIGMA1 method [34, 36] and the kinematics of elastic scattering [35]. Once these improvements were included in TARTNP the use of the code greatly expanded to include applications across the entire energy scale from thermal up to 20 MeV. Since then a wide variety of comparisons to other Monte Carlo codes, particularly codes that use continuous energy neutron cross sections, have demonstrated that TARTNP is: 1) very accurate, 2) incredibly fast - much of this speed being due to the use of the multi-band method, rather than continuous energy cross sections.

Over the last 20 years since 1975 TARTNP has been heavily used for a wide variety of applications, which has led to continuous feedback from users to improve the code and extend its capabilities to meet the needs of its many users. By working closely with TARTNP users and responding to their needs for an ever expanding variety of applications, today the code has been extended to include an extremely large menu of scoring and output options that meet the needs of most applications. Today using the combination of Doppler broadened cross sections to produce cross sections at virtually any temperature, thermal scattering kinematics, and multi-group cross sections in conjunction with the multi-band method to handle resonance self-shielding, and modern FORTRAN coding that allows TART95 to be used on virtually any computer (from CRAY super computers to IBM-PCs), TART95 is an extremely versatile code that can be used to very quickly solve a wide variety of problems.

Above we have discussed improvements in the physics of TART95 that have led to continuous improvements in the capabilities of the code. Here it is worth mentioning the incredible changes in the speed and size of computers that have contributed to allowing the code to be applied to ever more complicated problems. The early versions of SORS were run on a progression of computers including: IBM-7090, CDC-3600 and CDC-6600. Memory increased from 32 K to 128 K words, and speed increased by roughly a factor of ten, allowing SORS to run problems with up to 110 spatial zones. By the time TART came along it was used on CDC-7600, CRAY-1 and CRAY-2, memory varied from 256 K to 8,000 K words and again speed increased, allowing TART to run problems with up to 1,000 zones. This all involved the use of multi-million dollar super computers of their day. Today we are using inexpensive workstations that have up to 256 megabytes of memory, that run as fast as a CRAY-2, allowing us to run TART95 problems with up to 100,000 zones. For even less money you can run TART95 even on an IBM-PC at speeds comparable to a CRAY-1.

One point that has really contributed to simplifying the developing and maintenance of TARTNP during the last 20 years has been the decision to continue to use multi-group neutron data. This may sound like heresy today, when all codes seem to be using continuous energy cross sections. While continuous energy cross section Monte Carlo codes have been faced with the ever increasing size and detail included in modern neutron evaluation data libraries, TARTNP managed to keep plugging along using its 175 multi-group neutron structure and the multi-band method to handle resonance self-shielding, and yet a wide variety of comparisons indicate that the TARTNP results are as accurate as those obtained using continuous energy Monte Carlo codes. Over the years the only significant difference between TARTNP and continuous energy cross section codes has been the time required to calculate answers to within a given accuracy; this was true in 1975 and is even more so today as the ever increasing detail in modern neutron evaluations has continued to slow down Monte Carlo codes that use continuous energy cross sections. The past 20 years of experience indicates that the multi-band method could be called the ultimate and automatic variance reduction method for treating materials that have many resonances, e.g., U^{238}. It is the ultimate in the sense that it directly conserves important physical observables, such as average cross section and distance to collision, with just a hand full of histories. In comparison codes that use continuous energy cross sections must run an enormous number of histories in materials that have resonances, in order to obtain even a reasonable approximation to average cross sections and distance to collision. It is automatic in the sense that compared to other variance reduction methods that require the code user to be an expert in variance reduction in order to properly use it, with the multi-band method all the code user need specify by input is that the method should be used (**highly recommended**).

What Code Should You Use?

Do the above introductory remarks sounds like a commercial to encourage you to use TART95? Of course they are. But we should point out that before deciding whether or not you want to use it, you should ask the question: will it will meet your needs? Since you are the only one who really knows what your needs are, only you can answer this question. There are a few points that you should consider.

First, no computer code is perfect. We have run enough code intercomparisons to clearly demonstrate that today's computer codes are far too large and complicated for anyone to completely understand them, let alone guarantee that they are error free. Even with all the hard work in the world and the best of intentions we have to accept that ALL OF THEM contain errors. Without being able to intercompare results of exactly the same problem using a variety of codes it is impossible to guarantee the accuracy of the results of one code. If you would like to perform preliminary design calculations you can use any one of a variety of codes. But when you get to the point of finalizing calculations you should consider verifying your results using a variety of codes.

Second, no single computer is designed to produce the best answers most efficiently for all problems. In designing any code usually the designer has to compromise between the generality of the problems that a code can solve, and how efficiently, or fast, the code can solve problems. The results can be that a very general code may take too long to solve your problems, thereby making it impractical for your use. In the other extreme a very efficient, and fast, code may not be able to accurately solve your type of problem.

We deal with a wide variety of applications and for the above reasons we use a number of codes in combination to solve problems. We use TART95, as well as COG [32] and MCNP [33]. Having all three codes available for use allows is to: 1) verify the accuracy of results, and 2) select the code, or codes, that can best solve any given problem. Each of these codes has advantages and disadvantages that you should be aware of.

COG: has by far the most faithful and best representation of the available nuclear and atom data. It uses continuous energy, rather than multi-group, cross sections, and it uses continuous secondary angular and energy distributions exactly as they come from the evaluated data files. COG has a wide variety of user controlled variance reduction schemes. COG was originally designed to solve deep penetration problems and for this application a faithful representation of the secondary distributions is very important, as is the availability of variance reduction scheme. The disadvantage of COG is that of the three codes discussed here it is by far the slowest.

MCNP: is by far the most widely used of the codes discussed here. It also uses continuous energy, rather than multi-group, cross sections. The secondary angular and energy distributions are represented by equally probable bins, for speed of sampling. MCNP also has a wide variety of user controlled variance reduction schemes. MCNP is quite a general purpose code that can be used to solve a wide variety of problems; here the use of variance reduction can be used to great advantage by experienced users. The disadvantage of MCNP is also running time; it doesn't run anywhere nearly as slow as COG, but the continuous energy cross section treatments still requires a lot of computer time to reach convergence in many problems.

TART95: uses multi-group cross sections, and multi-band parameters to account for self-shielding. Like MCNP, the secondary angular and energy distributions are represented by equally probable bins, for speed of sampling. It has a minimum of user controlled variance reduction methods, and instead relies on its own build in expertise to accelerate problems. The major advantage of TART95 is its speed. Its major disadvantage would appear to be that it is not as general as either COG or MCNP, based on its treatment of nuclear and atomic data. But we feel that this has been more than offset by using the predictions of reactor theory to allow the code to accurately, accelerate to convergence in most problems, much faster than either of the other two codes.

As far as running time, comparison of a wide variety of results indicate that if we run COG, MCNP and TART95 on the same problem we can end up spending hours, minutes, or seconds, respectively using these codes.

We try to use the advantages of these codes by first running TART95 to perform as many preliminary calculations as possible; this allows us to complete these calculations as quickly as possible, or alternatively to perform many more calculations in the same amount of time to allow us to examine various possible designs. Periodically we verify our preliminary results by comparing TART95 results to those of the other codes. When it eventually comes to any final design study we are willing to spend more computer time and we run the codes in tandem to verify the final design.

Bottom line as far as what code should you use, is that we feel you should have a number of codes available for your use. Naturally we recommend that one of these codes be TART95.